Future Challenges to Linux Scalability

Andrew Morton
<akpm@linux-foundation.org>
<akpm@google.com>
Gelato ICE
April 2007

Current limits of kernel scalability

- 4MB of RAM
- 2MB might be possible
- Execution-in-place from Flash helps save RAM
- Improved modularization helps shrink kernel footprint
  - eg CONFIG_BLOCK
- But people say that 2.4 is still better
  - No specific reason - 2.6 has just got generally fatter
- Improved support for embedded CPUs, including MMU-less ones

A serious point

- Linux kernel presently scales from the very largest machines down quite small ones
- We are proud of this and want to maintain and even improve it
- This requires more care and discipline than would otherwise be the case
  - And cost

Another constraint on scalability work

- The other major constraint upon scalability work in the kernel: maintainability.
  - The kernel codebase will remain important and under development for decades
  - We want as many people as possible to understand it
    - To increase the number of effective developers
    - To minimise defect rates
- Scalability improvements sometimes require
  - complex algorithms
  - fancy lock types
  - complex data structures
  - hard-to-understand dynamic behavior

Another constraint on scalability work (cont'd)

- These complexity increases have costs:
  - higher likelihood of bugs
  - Decreased approachability
  - harder to add new features etc.
- All kernel merges involve consideration of cost/benefit trade-offs and sometimes a scalability improvement may not justify its increased long-term maintenance cost

Scalability: definition

- Achieve close-to-hardware performance across changes in
  - Number of disks, amount of memory, number of SMT threads, number of cores, number of CPUs, number of nodes, size of filesystems, number of files in a disk, directory, etc, number of locks on a file, number of NICs, number of open sockets, size of packets, number of disjoint mmap()s, network bandwidth, number of processes/threads, number of outstanding I/Os, request size, number of disks in an array, interrupt frequency, number of timers, number of open files, number of cached dentries/inodes, amount of swapcache, frequency of all syscalls, context-switch frequency, pagefault rate, amount of kernel statistical collection, futex acquisition frequency, number of fds in select()/poll(), select()/poll() wakeup frequency, ...
- Basically: everything
- For whichever conceivable bottleneck we have in the kernel, someone has a configuration or workload which will hit it

Scaling the kernel: an ongoing process

- We do a lot of work on scalability:
-
```
git-log | grep -i scalab | wc -l
```
- 107
- Scalability increases with maturity of the code
  - It takes time for scalability problems to appear, to be reported, to be understood and to be resolved
  - Often the machines/users who hit scaling bottlenecks are using ancient distro Enterprise kernels
    - The lag time before discovery of a scaling problem can be years
  - Lesson: it helps if people with large machines and challenging workloads test contemporary kernels
    - Can cut years out of the cycle

Scalability: current status

- 2.4 kernels tend to run out of steam at 4-CPU
- My rule-of-thumb during 2001-2004 was that any measurable scalability problem on 32-CPU machine was a bug
  - This was when we were still expecting that there would be a 2.7 kernel
- Scalability improvements continue to be merged at a constant rate
  - Although the problems which they fix are increasingly obscure
- We believe that current scalability is good:
  - terabytes of memory
  - hundreds of CPUs
  - thousands of disks

Scalability: expected problems

- Page reclaim
- Disk I/O
- Filesystems, filesystems, filesystems

Scalability: page reclaim

- Memory sizes are increasing
- Memory bandwidth is increasing
- Physical page size is not: still stuck at 4kbytes
  - Itanium can support up to 64k. 16k is apparently the sweet spot
  - Larger page sizes introduce filesystem inefficiencies with some workloads
- As memory size and speed increases, the fixed per-page processing in the VM becomes proportionally more expensive

Scalability: page reclaim (cont'd)

- OTOH, CPUs are getting faster too, so that per-page processing becomes faster
- So if memory size&speed increases outpace CPU speed increases, VM overhead becomes relatively larger
- Itanium can increase pagesize to amortize this, but x86 has no such option
- Hence for x86 scalability, we will need to hand-optimise these code paths
  - Itanium will benefit from that work as well

Scalability: increased disk IO bandwidth

- Similar to the page reclaim problem
  - as storage gets faster, per-page processing overhead becomes more costly
- If storage bandwidth growth outpaces CPU speedup, we might encounter problems
- But not a lot of hand-optimization has gone into the processing of pages for storage I/O. Yet

Filesystems: size

- The 16TB limit on ext3 will become an increasingly serious problem
- XFS has no such limit
- ext4 addresses the size problem

Filesystems: performance

- XFS is basically always good
  - excellent layout behavior
  - decent CPU efficiency
- ext4 is presently not as good in either department
  - but once all the presently-planned features are merged, ext4 will be competitive
  - IO performance for large directories might still be poor

Filesystems: XFS vs ext4

- Vendors are reluctant to support XFS
  - It is complex
  - Few kernel engineers understand it well enough to support it
  - Few companies support XFS
  - ext4 is better understood, is supported by engineers across the industry
- Distros seem to be converging on ext4 as the solution to our fs scalability problems

Filesystems: ext4

- The planned features for ext4 will address the size scalability problem
- Should largely address performance problems
- But ext4 development is slow

Summary

- Generally scalability appears to be adequate for today's machines and workloads
- Anticipated problems are mainly in in the memory-management and filesystem area
- The industry needs to ramp up investment in ext4 to head off a potentially serious performance crunch